Amendment to the Specification:

Please replace the paragraph starting on **page 3** with the following:

Existing computer systems are inherently incapable of providing sufficient security since the operating system that attempts to control and manage the processor exists as processor instructions; instructions that are functionally and operationally dependent on the same processor for their existence. The security problem is fundamental: the processor must execute instructions in order for the operating system to exist; the operating system must exist to control the very same processor that executes the instructions that are responsible for its existence...and round and round ~~we go~~it goes. This invention addresses the fundamental security problems that are responsible for data corruption in existing systems by presenting a new paradigm for computer systems: computer systems with an independently functioning and operating controller and resource management system and method, providing vital system-level security for the computer system.

Please replace the paragraph starting on **page 4** with the following:

Prior art systems are limited in their ability to identify and prevent unauthorized access and corruption of the shared system resources since the processor, memory and operating system are operatively and functionally linked together. Sharing system memory leaves prior art systems vulnerable to unauthorized accesses into application programs and operating system instructions. These unauthorized accesses lead to application errors, operating system instability, system lockups or persistent corruption of system resources. Furthermore, prior art operating systems and processors provide mutual and binding control over each other; the operating system attempts to control the processor, while the processor executes operating system instructions necessary for the operating system to control the very same processor. Problems are inevitable since the operating system and processor actually control each other; those skilled in the art will recognize that prior art operating systems do not independently provide control over the processor since it is impossible for the operating system to operate without having the processor

execute instructions necessary for the operating system to exist; the processor must execute software to allow the operating system to attempt to control the very same processor, all the while sharing the same memory space. ~~Does this make sense to Bill? No.~~

Please replace the paragraph starting on **page 30** with the following:

At startup the BIOS with device drivers 21 will boot the system and allow the processor 15 to begin loading and executing the prior art operating system 19. Those skilled in the art will realize that the processor is required to load and execute the instructions necessary for the prior art operating system to functionally operate. The processor's control over the prior art operating system is represented by arrow 16. The operating system is therefore dependent on the processor for its functional operation. The operating system, as a well-designed operating system should, attempts to control the processor as represented by arrow 17. Control arrows 16 and 17 help to illustrate the fact that the processor is therefore dependent on the operating system for its functional operation. As illustrated, prior art operating systems 19 used for providing control, management and security protection 22 for the entire computer system are mutually inclusive and conceptually, physically, functionally, operationally and electrically dependent of the processors 15, processor memory 14 and software 25-27. It should be noted that in the next drawing, FIG.8, control arrow 16 is conspicuously missing; it's no longer needed once ~~we separate~~ the function of the present invention is separated from the prior art processor. Control arrow 17 remains in FIG.8 since this arrow denotes the independent control that the present invention has over the processor. Even arrow 17 is dropped in later drawings since it's assumed that those skilled in the art will realize that the present invention's control over the processor happens to occur in-band as a result of bi-directional messaging in application program interface (API) path 40. Or alternately via secure interrupts to the processor 100.

Please replace the paragraph starting on **page 33** with the following:

Also shown in FIG. 8, the path taken 40 by the processor in the prior art block diagram of FIG. 7 has been drastically altered. The processor was able to get at the memory controller hub directly in prior art designs. As can be seen from FIG. 8, the processor is forced to go through the present invention in order to get to the memory controller hub as before. This is denoted by splitting the single arrow 40 of FIG.7 into three separate arrows 40 depicted in FIG.8. This is intentional by design since in alternate embodiments the processor will be forced to go through the security function of the present invention. The general computer system resources such as the memory controller hub 9 used by the processor and other system resources to arbitrate for access to the shared system bus 23 and shared system resources 2 are also shown along with high-speed video interconnect 38 and Gigabit Ethernet (GbE) interface 37 and PCI bus resources 33. Interfaces 41 and 42 have been added to [[our]]the present invention 1. Interfaces 41 are used to operatively and communicably coupling separate present invention controller and resource management system systems together that reside in the same computer system. Interfaces 42 are used to operatively and communicably coupling separate present invention controller and resource management system together that happen to be in separate computer systems. Interfaces 41 and 42 provide the computer systems a means to directly couple present invention controller and resource management system together independent of the processing function. This provides improvements in security and reliability over prior art systems that instead couple the processors together, leaving the entire computer system, processor, prior art operating system and application programs vulnerable to corruption. Since the controlling and managing function for the entire computer system is now independently controlling the entire computer system, it just makes sense to tie the controlling functions together directly rather than going through the processing function. This is impossible to do with prior art systems since the operating system and processing function are mutually dependent functions.

Please replace the paragraph starting on **page 43** with the following:

The basic input output system (BIOS) with device drivers 21, the device configuration manager 44 and device configuration table 58 (alternately stored in external memory) are provided primarily for booting or updating the computer system via interfaces 23, 40, 42, 41, 29-35 or 37. The processor and shared system memory no longer have to get involved in booting or configuring the system, or interfacing with computer system resources via device drivers; no software is required. The invention is an improvement over prior art since booting will happen quicker and also be protected from unauthorized accesses, corruption or application program errors; [[our]]the system is therefore more reliable, stable, secure and higher-performing when compared to prior art computer systems.

Please replace the paragraph starting on **page 48** with the following:

Those skilled in the art will realize that an example of a fully functional computer system operating independently of the processor and processor instructions can be demonstrated using the embodiment 12 of the present invention in conjunction with high level flow diagrams of FIGs. 17-19. The present invention 12 will begin initial operations after receiving a power on reset event 68; the functions primarily responsible for booting the computer system and configuring the computer system devices are functions: clocks 61, BIOS and device drivers 21, device configuration table 58 and device configuration manager 44 (keeping in mind that other functions of the present invention 12 are required to support the booting and configuring operations; they are also required to execute low priority background tasks.) Once booting and configuring are complete, the present invention 12 is now prepared to respond to computer system events 71, initiated on interfaces 23, 40, 29-35, 37, 38 41 or 42 consisting of; interrupts, received messages or state changes in status buffers. ~~Let's say that we happen to receive~~Assume an interrupt is received from keyboard interface 35 indicating that a local user of the computer system has input a text message to be sent out another interface 29; the present invention must also send this very same message to the user interface (video monitor) 38. The event handlers

66, 67 along with the resource scheduler 19 and system security function 6 will categorize the keyboard event and proceed to place a "red", "yellow" or "green" tag to the event. ~~Let's assume~~Assume the keyboard event gets a "green" tag; since keyboard entries are very slow events that are buffered, the present invention may want to finish off some background processes 70 while waiting for the high-water threshold of the keyboard buffer before starting to service the keyboard buffer. When the system has determined that it is time to service the keyboard it will forward all data to the system security function 22 via keyboard interface 35, I/O controller hub 36 and internal bus 47. Each keystroke is interrogated by the system security function 6 in order to flag unauthorized attempts to access computer system resources via the keyboard interface 35. ~~Let's assume that~~Assume a complete text message was entered and some of the data has been flagged "very-bright-red", (a certain four keys were mischievously pressed). [[Our]]The system security protection can choose not to display these four keys back to the user via the video monitor. The remaining "green-flagged" data will be displayed on the video monitor. Meanwhile the data has been temporarily buffered in computer system memory 25 via memory controller 39 and memory interface 23. If the present invention detects a button "click" event on mouse I/F 34 it may respond by retrieving the stored data from memory 25 via memory I/F 23 and memory controller 39. The present invention can also choose to re-verify the data using the system security function 6 depending on how "aged" the data is. Since the data was assigned type and security level identifier labels when previously stored in computer system memory 25, those identifiers can now be read in order to determine the proper destination(s) for the data. The data can now be forwarded to any computer system interface, as well as broadcast or multicast out multiple interfaces if need be. ~~Let's assume that~~Assume this text message is intended to be sent to a local printer, to a friend on a local area network(LAN)29, to a video game executing as another process on this same computer system and also to a text-to-speech interface just for fun (it's good ~~that we were~~ the present invention was able to previously filter those four mischievous keys just in case the volume is cranked up on the text-to speech audio interface 30). The present invention is fully capable of broadcasting, (or more likely in this case multicasting) to multiple destinations. In this example, ~~we have already sent~~ the keyed data has already been sent to the user interface via 38; it can now be multicast to the local printer attached to either serial input/output (SIO)32 interface or universal serial bus (USB)31. The data is also multicast to

local area network (LAN) interface 29 via layer-2 media access controller (MAC) 46 integrated into I/O controller hub 36; the "friend" receives the eagerly awaited text message (sans the missing four keys); meanwhile the text-to-speech message has been sent out coder/decoder (CODEC) interface 30 for the long awaited audio announcement (again, sans four key letters). This example is intended to highlight some of the unique functions and features integral to this invention: system-level events can take place independently of the computer system processor, multi-level security is available at every interface and in every direction within the present invention, data can be multicast or broadcast out multiple computer system interfaces.